4

Holmdel
Lyons, Bob
◀ USA ▶
Unidex Inc.

Bob Lyons

Electronic Commerce Consultant

Unidex Inc.

8 Stoecker Road Holmdel (New Jersey) USA (07733)
Email: boblyons@unidex.com **Web site:**http://www.unidex.com/

Biography

Electronic
Commerce
Consultant
XML Convert

Bob Lyons is an electronic commerce consultant with Unidex Inc., where he helps clients develop electronic commerce solutions, including extranet applications and EDI servers. Bob developed XML Convert, which is a free tool that converts flat file data to XML documents and vice versa. He has over 12 years of electronic commerce experience. Bob has led electronic commerce implementations at large corporations, given numerous presentations on electronic commerce, and has published articles on EDI, the Internet and X.400.

Introduction

◀ Convert Flat File to XML ▶
◀ Convert XML to Flat File ▶
◀ Export Flat File ▶
◀ Import Flat File ▶

Companies are beginning to use XML to send application data to browsers and to business applications. XML is well suited for the interchange of data, since XML documents are self-describing, easily parsed and can represent complex data structures. Also, there is a wide variety of high-quality, inexpensive tools for parsing and transforming XML documents. When using XML for data interchange, ideally, the sending application will be able to export an XML document, and the receiving application will be able to import an XML document. Unfortunately, many legacy applications use flat files to import or export data. So, companies will need to convert flat files into XML documents when sending data to XML-capable applications. Likewise, companies will need to convert XML documents into flat files that can be imported into legacy applications.

Flat Files

◀ Field ▶
Flat Files
◀ Record ▶

Flat files contain machine-readable data that is typically encoded as printable characters. A flat file usually contains a series of records (or lines), where each record is a sequence of fields. A field contains an atomic piece of data (e.g., a postal code).

◀ Example Flat File ▶

Let's look at a simple flat file containing employee data. The file contains one or more employee records. Each record contains the following three fields:

- Employee's social security number (ssn)
- Employee's full name (last name followed by a comma followed by a space followed by the first name)

- Employee's salary

The following are the contents of the employees flat file:

```
123456789,"Carr, Lisa",100000.00
444556666,"Barr, Clark",87000.00
777227878,"Rabbitt, Jack",123000.00
```

◀ CSV ▶ Comma Separated Value ◀ Line Separator ▶ ◀ Quotes ▶

Each record contains information about one employee. The format of the flat file is Comma Separated Value (CSV), which means that each record is terminated by the operating system's line separator and the fields within a record are separated by a comma. In addition, a field value may be enclosed in quotes, which escape any commas or line terminator characters that appear within the field value. Note that the quotes that surround the field value are not actually part of the field value. Also, if a field value contains a quote character, then the field value must be surrounded by quotes and the quote character in the field value must escaped by prefixing it with an additional quote.

◀ Example Flat File ▶

Let's look at a more complicated flat file, the structure of which is similar to the structure of a Windows initialization file. The flat file contains a list of contacts. The following are the contents of the contacts flat file:

```
[contact]
name: Nancy Magill
email: lil.magill@blackmountainhills.com
phone: (100) 555-9328
[contact]
email: molly.jones@oblada.com
name: Molly Jones
[contact]
phone: (200) 555-3249
name: Penny Lane
email: plane@bluesuburbanskies.com
```

Each contact consists of a begin contact record followed by three optional records. The begin contact record consists of the string "[contact]". The three optional records, which can appear in any order, are as follows:

- The name record, which contains the full name of the contact. This record begins with a "name=" label followed by the name.
- The email record, which contains the email address of the contact. This record begins with a "email=" label followed by the email address.
- The phone record, which contains the phone number of the contact. This record begins with a "phone=" label followed by the phone number.

◀ Line Separator ▶

Each record in the flat file is a line that is terminated with the operating system's line separator.

Flat Hierarchical

You might be wondering why these two files are considered "flat". The term "flat" means that the file is not indexed. The term also implies that a flat file does not have a hierarchical structure; however, many flat files do have a hierarchical

◀ Nested Groups of Records ▶

structure. Even simple flat files, such as the employees file above, contain a sequence of records where each record contains a sequence of fields. Many flat files, such as those used to exchange insurance claims, have more complicated data structures, such as multiple record types, groups of records, nested groups of records, repeating groups, etc.

◀ Export Flat File ▶
◀ Import Flat File ▶

Flat files are commonly used to transfer data between two applications, since many business applications (e.g., SAP's R/3, EDI translators, legacy applications, etc.) use flat files to import and export data. For example, when a company receives an EDI invoice from a vendor, it will use an EDI translator to convert the invoice data from the EDI data format (e.g., X12) into the data format required by the accounts payable system. The EDI translator will produce a flat file containing the converted invoice data. The accounts payable system then imports this flat file.

◀ Conversion Tools ▶
Export XML
Import XML

In the future, many business applications will be able to import and export XML. For example, SAP has announced that R/3 will be able to import and export XML, in addition to importing and exporting the SAP IDOC flat files. Until then, there will be a need for conversion tools that can convert complex flat files into XML documents, and vice versa.

Conversion Between Flat Files and XML

◀ Convert Flat File to XML ▶

Companies will need to convert flat files to XML when transferring data from legacy applications to XML-capable applications (e.g., SAP's R/3, Microsoft's IE 5.0, etc.). Companies will also convert flat files to XML when they need to display the flat file data on a non-XML-capable browser, since it is easy to convert XML to HTML using XSLT.

◀ Convert XML to Flat File ▶

Companies will need to convert XML into flat files when transferring data from XML-capable systems to a legacy system.

◀ Conversion Tools ▶
Schema-Driven

Conversion between flat file data and XML can be done via generic conversion tools or custom scripts. Generic conversion tools are schema-driven, so that they can handle a wide range of flat file formats. Such a conversion tool uses the schema of the flat file in order to parse the file and convert it to an XML document. The conversion tool also needs the flat file schema when converting an XML document into a flat file that conforms to the flat file schema.

The XFlat Language

XFlat
XFlat Instance
XFlat Schema

XFlat is an XML language for defining flat file schemas. An XFlat schema is an XML document that conforms to the XFlat language and that describes the format of a flat file. An XFlat schema defines the structure and syntax of a flat file that contains non-XML data. An XFlat schema also defines the structure and syntax of anXFlat instance . An XFlat instance is an XML document whose structure is the same as the flat file and whose data is the same as the data in the flat file. In other words, an XFlat schema describes the structure of a flat file and the corresponding XFlat instance.

Delimited
◀ Field ▶
◀ Field Separator ▶
Fixed length
◀ Record ▶

The flat file that is described by an XFlat schema must consist of records, where each record is a sequence of fields. A field is an atomic piece of data (e.g., a postal code). Records and fields may be delimited. A record separator (i.e., delimiter) occurs at the end of a record and helps the parser determine where the record ends. Likewise, a field separator occurs at the end of a field and helps a parser to

◀ Record Separator ▶

determine where a field ends. Fields that are not delimited must be fixed length (i.e., the minimum length of the field must be equal to the maximum length of the field).

◀ Nested Groups of Records ▶ Recursion Recursive Data Structures Subgroups

The records may be grouped, and groups of records may be nested in a hierarchical structure (in other words, groups of records may contain subgroups). Note that XFlat does not support recursive data structures.

XFlat Element Types

The XFlat element types are as follows:

- XFlat Element

  XFlat, which is used to define an XFlat schema. The XFlat element is always the document element (i.e., root element) of an XFlat schema. An XFlat element must contain exactly one subelement, and this subelement must be a SequenceDef element, a ChoiceDef element or a RecordDef element.

- SequenceDef Element

  SequenceDef, which is used to define a sequence of objects, where each object may be a sequence, a choice or a record. A SequenceDef element must contain one or more subelements; each of these subelements must be a SequenceDef element, a ChoiceDef element or a RecordDef element.

- ChoiceDef Element

  ChoiceDef, which is used to define a choice of one of a set of objects, where each object may be a choice, a sequence or a record. A ChoiceDef element must contain one or more subelements; each of these subelements must be a SequenceDef element, a ChoiceDef element or a RecordDef element.

- RecordDef Element

  RecordDef, which is used to define a record, which is essentially a sequence of fields. A RecordDef element must contain one or more FieldDef elements.

- ◀ FieldDef Element ▶

  FieldDef, which is used to define a field. A field contains an atomic piece of data. A FieldDef element may not contain any subelements.

◀ MapToXml Attribute ▶

An XFlat schema contains all the information needed to convert a flat file to XML (or vice versa). The MapToXml attribute in the XFlat language allows you to map each group, record and field to an XML element or to nothing. A field can also be mapped to an XML attribute.

Declarative Language

Note that XFlat is a declarative language. A non-programmer who is familiar with flat files can create an XFlat schema.

Download XML Convert
◀ For More Information ▶
◀ http://www.unidex.com/ ▶

For more information about XFlat (e.g., definitions of the XML attributes in the XFlat language), please download XML Convert at http://www.unidex.com/ and see the documentation that accompanies the application.

Example XFlat Schemas

Let's look at the XFlat schema for the employees flat file. The contents of that file were as follows:

```
123456789,"Carr, Lisa",100000.00
444556666,"Barr, Clark",87000.00
777227878,"Rabbitt, Jack",123000.00
```

◀ Example XFlat Schema ▶

The following XFlat schema describes the layout of the employees flat file:

```
<XFlat Name="employees_schema" Description="CSV flat file">
    <SequenceDef Name="employees" Description="employees flat file">
        <RecordDef Name="employee" FieldSep="," RecSep="\\N" MaxOccur="0">
            <FieldDef Name="ssn" NullAllowed="No"
                         MinFieldLength="9" MaxFieldLength="9"
                         DataType="Integer" MinValue="0"
                         QuotedValue="Yes"/>
            <FieldDef Name="name" NullAllowed="No"
                         QuotedValue="Yes"/>
            <FieldDef Name="salary" NullAllowed="No"
                         DataType="Float" MinValue="0"
                         QuotedValue="Yes"/>
        </RecordDef>
    </SequenceDef>
</XFlat>
```

Please note the following about this XFlat schema:

- Name Attribute — Each of the following is mapped to an XML element in the XFlat instance by default: the SequenceDef element for the flat file, the RecordDef for the employee record and the FieldDef elements for the three fields. The tags for the XFlat instance are specified by the Name attributes in the SequenceDef, ChoiceDef, RecordDef and FieldDef elements.
- MaxOccur Attribute — The RecordDef element contains the MaxOccur="0" attribute, which means that there is no upper limit on the number of the employee records that may appear in the flat file.
- Description Attribute — The Description attribute, which is used in the XFlat, SequenceDef and RecordDef elements, contains free form text and is treated as a comment.
- Attribute — The record separator for the record is defined as "\\N", which is

Normalization ◀ Line Separator ▶ RecSep Attribute ◀ Record Separator ▶ ◀ XFlat Encoding ▶	an XFlat encoding for the line separator for the local operating system. If we were to define the value of the RecSep attribute as "&#D;&#A;", then an XML parser would convert this value to a space character during the attribute normalization process. XFlat has encodings for a handful of special characters that are affected by attribute normalization. On Unix, "\\N" is converted to the line feed character (Unicode #xA). On Windows, "\\N" is converted to the carriage return character (Unicode #xD) followed by the line feed character (Unicode #xA).
• ◀ CSV ▶ ◀ FieldDef Element ▶ FieldSep Attribute QuotedValue Attribute ◀ Quotes ▶	All the FieldDef elements contain the QuotedValue="Yes" attribute, since the flat file format is CSV. For the same reason, the RecordDef element contains the FieldSep="," attribute.
• NullAllowed Attribute	All three FieldDef elements contain the NullAllowed="No" attribute, since all three fields are mandatory (i.e., the length of the field value must be greater than or equal to one character).
• Data Type	The data type of the Social Security Number (ssn) field is declared as "Integer".
• Minimum Value	The minimum value of the Social Security Number (ssn) field is set to zero, since a negative social security number is invalid.
•	The data type of the salary field is declared as "Float".
•	The minimum value of the Salary field is set to zero, since a negative salary is invalid.
• MaxFieldLength Attribute	The FieldDef elements for the name and salary fields do not include the MaxFieldLength attribute. Thus, the default maximum field length (i.e., 80 characters) applies to both fields.

Now let's look at the XFlat schema for the contacts flat file. The following were the contents of that file:

```
[contact]
name: Nancy Magill
email: lil.magill@blackmountainhills.com
phone: (100) 555-9328
[contact]
email: molly.jones@oblada.com
name: Molly Jones
[contact]
```

```
phone: (200) 555-3249
name: Penny Lane
email: plane@bluesuburbanskies.com
```

◀ Example XFlat Schema ▶

The following XFlat schema describes the layout of the contacts flat file:

```
<XFlat Name="contacts_schema" Description="unordered records">
    <SequenceDef Name="contacts">
        <SequenceDef Name="contact" MinOccur="0" MaxOccur="0">
            <RecordDef Name="begin_contact" MapToXml="No" RecSep="\\N">
                <FieldDef Name="begin_contact"
                          ValidValue="[contact]" MapToXml="No"/>
            </RecordDef>
            <ChoiceDef Name="unordered_recs" MapToXml="No"
                       MinOccur="0" MaxOccur="3">
                <RecordDef Name="full_name" RecSep="\\N" MapToXml="No">
                    <FieldDef Name="label"
                              MinFieldLength="5" MaxFieldLength="5"
                              ValidValue="name=" MapToXml="No"/>
                    <FieldDef Name="full_name"/>
                </RecordDef>
                <RecordDef Name="phone_num" RecSep="\\N" MapToXml="No">
                    <FieldDef Name="label"
                              MinFieldLength="6" MaxFieldLength="6"
                              ValidValue="phone=" MapToXml="No"/>
                    <FieldDef Name="phone_number"/>
                </RecordDef>
                <RecordDef Name="email" RecSep="\\N" MapToXml="No">
                    <FieldDef Name="label"
                              MinFieldLength="6" MaxFieldLength="6"
                              ValidValue="email=" MapToXml="No"/>
                    <FieldDef Name="email_address"/>
                </RecordDef>
            </ChoiceDef>
        </SequenceDef>
    </SequenceDef>
</XFlat>
```

Please note the following about this XFlat schema:

- ◀ Line Separator ▶ ◀ Record Separator ▶ ◀ XFlat Encoding ▶ The record separator for the record is defined as "\\N", which is an XFlat encoding for the line separator for the local operating system.

- ◀ MapToXml Attribute ▶ The FieldDef elements for all the label fields contain the MapToXml="No" attribute, since we don't want to map the label fields to XML.

- ◀ MapToXml All four RecordDef elements contain the MapToXml="No"

Attribute ▷

element, since there is only one interesting field in each of these four record types.

- ◀ Field Separator ▷
  Fixed Length Field
  ◀ Record Separator ▷
  Variable Length Field

  Each of the RecordDef elements within the ChoiceDef element consists of a fixed length field (with no field separator) followed by a variable length field that is delimited by the record separator.

- This schema does not describe all the syntax rules for this flat file. For example, a contact is not allowed to contain two name records; unfortunately, the XFlat language is not yet capable of expressing such a syntactical constraint.

XML Convert

◀ Convert Flat File to XML ▷
◀ Convert XML to Flat File ▷
◀ Java Application ▷
Next Version

XML Convert is a free Java application that uses XFlat schemas to convert flat files into XML, and vice versa. XML Convert 1.1, which is the version currently available at http://www.unidex.com/, converts flat files into XML, but does not convert XML into flat files. The next version, which converts in both directions and which is described in this paper, will be available shortly at http://www.unidex.com/.

Key Features

The key features of XML Convert include:

- Fixed Length
  Human-Readable Report
  Semi-Structured Data
  Variable Length
  Wide Range of Flat File Formats

  Handles a wide range of flat file formats (e.g., variable length records, CSV, fixed length records, files that contain multiple record types, nested groups of records, records that contain a mixed of delimited and non-delimited fields, records in which each field has a different delimiter, etc.). XML Convert can even handle semi-structured data, such as a human-readable report that contains rows and columns of data.

- XML Language

  Uses a simple XML language (i.e., XFlat) for the flat file schemas.

- Portability

  Written in Java for portability.

- ◀ Java

Application ▷ Windows Executable	Includes a Windows executable for ease of use on a PC. XML Convert can also be invoked as a Java application from the command line.
•	
◀ API, Application Programming Interface ▷	Includes an API, so that XML Convert can be easily invoked from the user's Java application.
•	
OutputDocumentHandler Interface ◀ XSLT Stylesheet ▷ ◀ XT ▷ ◀ XflatOutputHandler Class ▷	Includes the com.unidex.xflat.XflatOutputHandler class that implements the com.jclark.xsl.sax.OutputDocumentHandler interface. When the com.unidex.xflat.XflatOutputHandler class is specified as the output method in an XSLT stylesheet, XT will pass the result tree to XML Convert, which will validate the result tree against the XFlat schema and, if the result tree conforms to the XFlat schema, produce a flat file.
•	
◀ Validation of Data ▷	Validates the flat file or XFlat instance against the XFlat schema. If an error is found in the input file (i.e., the flat file or the XFlat instance), then the conversion process is terminated and a detailed error message is generated.
•	
Error Messages	Error messages include a detailed description of the error and the location of the error (within the XFlat schema file, the XFlat instance or the flat file), so that the user can quickly troubleshoot the error.
•	
Large Flat Files Stream Mode	Converts in stream (i.e., serial) mode, which means that XML Convert does not read the entire flat file or XFlat instance into memory. Thus, XML Convert can convert very large flat files into XML, and vice versa.
•	
	XML Convert is free.

◀ Convert Flat File to XML ▷ ◀ Validation of Data ▷

When XML Convert transforms a flat file to an XML document (i.e., an XFlat instance), it will verify the structure of the flat file data and the data types of the fields using the XFlat schema. If the flat file does not pass this verification, then it is rejected. This verification minimizes the chance that an invalid XML document will be sent to the receiving application.

◀ Convert XML to Flat File ▷

Likewise, when XML Convert transforms an XML document to a flat file, it will verify that the resulting flat file conforms with the XFlat schema. This verification minimizes the chance that an invalid flat file will be imported into a business application.

Converting Between the Employees Flat File and XML

◀ Convert Flat File to XML ▷

Using the XFlat schema for the employees flat file (see above), XML Convert would convert the employees flat file into the following XML document (i.e., XFlat instance):

```
<?xml version='1.0'?>
<employees>
    <employee>
        <ssn>123456789</ssn>
        <name>Carr, Lisa</name>
        <salary>100000.00</salary>
    </employee>
    <employee>
        <ssn>444556666</ssn>
        <name>Barr, Clark</name>
        <salary>87000.00</salary>
    </employee>
    <employee>
        <ssn>777227878</ssn>
        <name>Rabbitt, Jack</name>
        <salary>123000.00</salary>
    </employee>
</employees>
```

◀ Convert XML to Flat File ▷

In the reverse direction, using the same XFlat schema, XML Convert would convert this XFlat instance back into the original employees flat file.

Converting Between the Contacts Flat File and XML

◀ Convert Flat File to XML ▷

Using the XFlat schema for the contacts flat file (see above), XML Convert would convert the contacts flat file into the following XML document:

```
<?xml version='1.0'?>
<contacts>
    <contact>
        <full_name>Nancy Magill</full_name>
        <email_address>lil.magill@blackmountainhills.com</email_address>
        <phone_number>(100) 555-9328</phone_number>
    </contact>
    <contact>
        <email_address>molly.jones@oblada.com</email_address>
        <full_name>Molly Jones</full_name>
    </contact>
    <contact>
        <phone_number>(200) 555-3249</phone_number>
        <full_name>Penny Lane</full_name>
        <email_address>plane@bluesuburbanskies.com</email_address>
    </contact>
</contacts>
```

◀ Convert XML to Flat File ▷

In the reverse direction, using the same XFlat schema, XML Convert would convert this XFlat instance back into the original contacts flat file.

Using XSLT With XML Convert

After converting a flat file into XML using XML Convert, it may be necessary to change the structure and/or element names of the resulting XML document (i.e.,

◀ Convert XML to Flat File ▶
◀ XSLT ▶
◀ XSLT Processor ▶

the XFlat instance) before sending it to the receiving application. For example, if the resulting XFlat instance will be sent to a browser that does not support XML, then the XFlat instance should be converted from XML to HTML using an XSLT processor. If the output will be sent to an XML-capable application, then it will probably be necessary to use an XSLT processor to convert the XFlat instance into a new XML document whose structure meets the requirements of the receiving application. (Note that the output of the XSLT processor can be an XML document or an HTML document.)

If the resulting XFlat instance will be sent to an XML-capable browser, then the XFlat instance can specify a stylesheet, so that the browser renders the XML document as a nicely formatted web page.

◀ Convert XML to Flat File ▶

When converting an XML document into a flat file, the XML document would probably not have the same structure as the target flat file. In this case, the user can use an XSLT processor to convert the XML document into an XFlat instance (i.e., an XML document whose structure is the same as the structure of the target flat file). The user would then employ XML Convert to transform the XFlat instance into a flat file. XML Convert uses an XFlat schema to parse the XFlat instance and produce the target flat file.

XSLT Output Method
◀ XT ▶
◀ XflatOutputHandler Class ▶

If you are using XT to convert the XML document into an XFlat instance, then the XSLT stylesheet can specify the com.unidex.xflat.XflatOutputHandler class as the output method, so that XT passes the result tree to XML Convert, which will validate the result tree against the XFlat schema and, if the result tree conforms to the XFlat schema, produce a flat file.

◀ XSLT Stylesheet ▶

The following is an example of an XSLT stylesheet that specifies the com.unidex.xflat.XflatOutputHandler class as the output method:

```
<xsl:stylesheet
  xmlns:xsl="http://www.w3.org/XSL/Transform/1.0">
    <xsl:output method="xtj:com.unidex.xflat.XflatOutputHandler"
                xmlns:xtj="http://www.jclark.com/xt/java"/>
    <!-- The rest of the stylesheet goes here. -->
</xsl:stylesheet>
```

Transformation Features

Note that XML Convert and the XFlat language do not provide any XML to XML transformation features, since XSLT can be used to do XML to XML transformation.

◀ CSV ▶
Field Length Fields
◀ XSLT Processor ▶

Also note that an XSLT processor can convert an XML document into non-XML text, without any help from XML Convert. Thus, you could use an XSLT processor without XML Convert to transform an XML document into a flat file. However, it would be very difficult to write an XSLT stylesheet that syntactically validates the resulting flat file. It's important to validate the resulting flat file, so that the receiving application does not import an invalid flat file. Also, it would be difficult to write a stylesheet that produces a CSV flat file, since the stylesheet would have to escape any quote characters that are embedded in the field values. It would also be difficult to write a stylesheet that produces a flat file containing fixed length fields, since the stylesheet would have to pad the values of some fields with spaces, so that the length of each field is correct.

Summary

XML Convert is a free Java application that uses XFlat schemas to convert flat files into XML and vice versa. XFlat is an XML language for defining flat file schemas. XML Convert uses an XFlat schema to parse and validate the input file (i.e., the flat file or the XFlat instance), and to produce the output file. XML Convert supports a wide variety of flat file formats, including CSV, semi-structured data (e.g., human readable reports), fixed length records and fields, multiple record types, groups of records, nested groups, etc.

◀ For More Information ▶

◀ http://www.unidex.com/ ▶

For more information about XML Convert and XFlat, please see http://www.unidex.com/.

Smarting up your legacy data | Table of contents | Indexes | Enterprise Information Portals